International Journal of Computer Applications (0975 – 8887)
Volume 127 – No.16, October 2015

CAN based Simulator Design for Vehicles

Rohini Shinde
M.E.S College of Engineering, Pune, Maharashtra, India

Manisha Dale
M.E.S College of Engineering, Pune, Maharashtra, India

Gajanan Udas
Dept. of Advanced Technology, TATA Technologies, Pune, Maharashtra, India

ABSTRACT

Control Area Network (CAN) based Simulator is developed with help of LabVIEW platform which simulates the electrical system inside the vehicle. This simulator is used to test the android application running on the dashboard of a vehicle. Entire design is based on the CAN architecture, its messaging system and functionality of the electronic control units (ECUs) inside vehicles. Testing depth increased successfully with proposed simulator.

General Terms

Embedded Systems, Communication and Networking, Vehicle Transport Systems

Keywords

CAN, ECU, Messaging Protocol, Reception of frames, and Transmission of frames.

1. INTRODUCTION

Every vehicle is having the dashboard. It is a control panel placed in front of the driver of an automobile, which controls the operation of the vehicle. The instrument cluster contains gauges such as a speedometer, tachometer, odometer and fuel gauge and indicators such as gearshift position. Electronic control unit (ECU) is an embedded system that controls one or more of the electrical systems in a motor vehicle. ECU reads signal coming from sensors placed at various parts of the vehicle and depending on this information, it controls various important units for example engine and automated operations within the car. It also keeps a check on the performance of some key components used in the vehicle. As the comfort level in vehicle increases the number of ECUs gets increased due to increased level of automation. Therefore the wiring harness also gets complicated when we go for point to point communication of ECUs. With the use of CAN serial bus wiring harness get reduced. The CAN bus standard was introduced by the Robert Bosch as communication protocol in vehicle network. The CAN is a multi-master serial bus used for connecting several ECUs and dashboards [2, 3].

In order to make the instrument panel fully digital many manufacturer are trying to build an android applications which will indicate all parameters of the vehicle. Every time it is not possible to test such application on field, by interfacing it with vehicle. On field testing in development phase of digital dashboard is very time consuming and costly method. Hence it gives rise to the need of development of simulator that will give the same environment in vehicle, which provides information coming from ECUs in the form of message packets of CAN protocol.

Proposed Simulator design is entirely based on the CAN 2.0B standard of messaging protocol which consists of standard frame (11 bit identifier) and extended frame (29 bit identifier)[1]. This paper consists of sections as follows: CAN protocol is explained in section 2, while the proposed methodology of simulator design and its work flow is mentioned in section 3. Experimental results of the simulator are given in section 4 which gives the idea about all tests.

2. CAN MESSAGING PROTOCOL

CAN is peer to peer network protocol. It provides maximum speed up to 1Mbps. CAN cover the lower two layers of OSI model, which is data link and physical layer respectively. As shown in Figure 1 individual node in network have access to read and write data on the CAN bus.

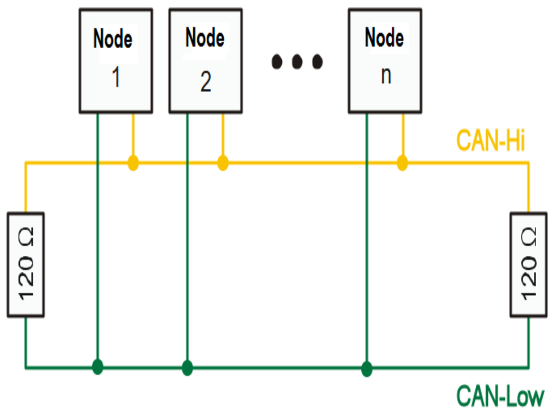


Figure 1: Control Area Network

CAN node may transmit and receive message frames but not simultaneously onto the network. An arbitration ID is unique throughout the network which labels the frame. If multiple nodes try to transmit a message onto the CAN bus at the same time, the node with the highest priority (lowest arbitration ID) automatically gets bus access. Lower priority nodes must wait until the bus becomes available before trying to transmit again. In this way, we can implement CAN networks to ensure deterministic communication among CAN nodes. Termination resistor of value 120Ω is normally used to activate the CAN bus [1].

CAN specification 2.0B consist of standard frame and the extended frame format as shown in Figure 2 and Figure 3. Both the frames are almost same except the identifier field. Identifier extension (IDE) bit indicate whether it is standard or extended. Remote Transmission Request (RTR) differentiates between the data frame and remote frame.

S O F	11 Bit Identifier	R T R	I D E	R0	D L C	0 to 8 Byte Data	C R C	A C K	E O F	I F S

Figure 2: Standard frame (11 bit Identifier)

SOF	11 Bit Identifier	SRR	IDE	18 Bit Identifier	RTR	r1	r0	DLC	0 To 8 Bytes Data	CRC	ACK	EOF	IFS

Figure 3: Extended Frame (29 bit Identifier)

If RTR is dominant bit (logic 0) it indicates it is a data frame otherwise remote frame. Substitute Remote Request (SRR) field which replaces the RTR in standard frame as placeholder in extended frame. Four bits of Data Length Code (DLC) decides the number of data bytes present in data field. Cyclic Redundancy Check (CRC) occupies 15 bits plus delimiter. Acknowledgement (ACK) consists of 2 bits which get overwritten by the receiving node with dominant bit, indicating an error free message has been sent. Start of the frame (SOF) uses dominant bit to assure the beginning of the frame and end of the frame (EOF) field utilizes 7 bits. To separate out two frames inter frame space (IFS) with 7 bits is used. The proposed simulator design is based on these frame structures. Here the nodes are nothing but the ECUs and the data field of frame consist of the sensors data such as engine temperature, coolant temperature, engine speed, odometer, battery voltage etc[1]-[3].

3. PROPOSED METHODOLOGY

In automobile test systems, it is essential to take account of the test cost, efficiency, safety, duration and feasibility. Hence the development of simulator should be good enough so that it exactly replicate the communication among the ECUs and dashboard. The simulator should be efficiently designed in order to enhance the test quality. Figure 4 shows the concept of simulator used for testing of android application running on the dashboard tablet.

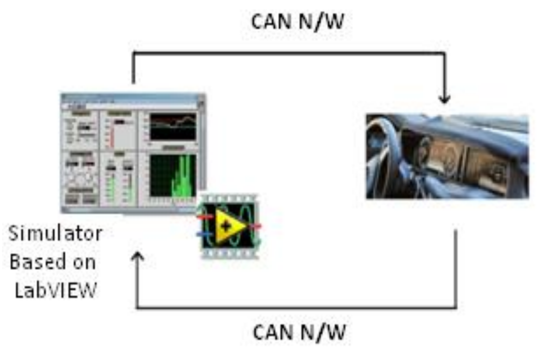


Figure 4: Concept of proposed simulator.

3.1 Proposed Architecture

The proposed simulator designing process is done in two modes. First mode consists of Message frame transmission that means writing of ECU messages from simulator to android application. The second mode is request frame reception that includes reading of request coming from the android application to the simulator and then sending its acknowledgement signal along with its appropriate response again to the application as shown in Figure 4. The detailed architecture of the proposed simulator is shown in Figure 5.

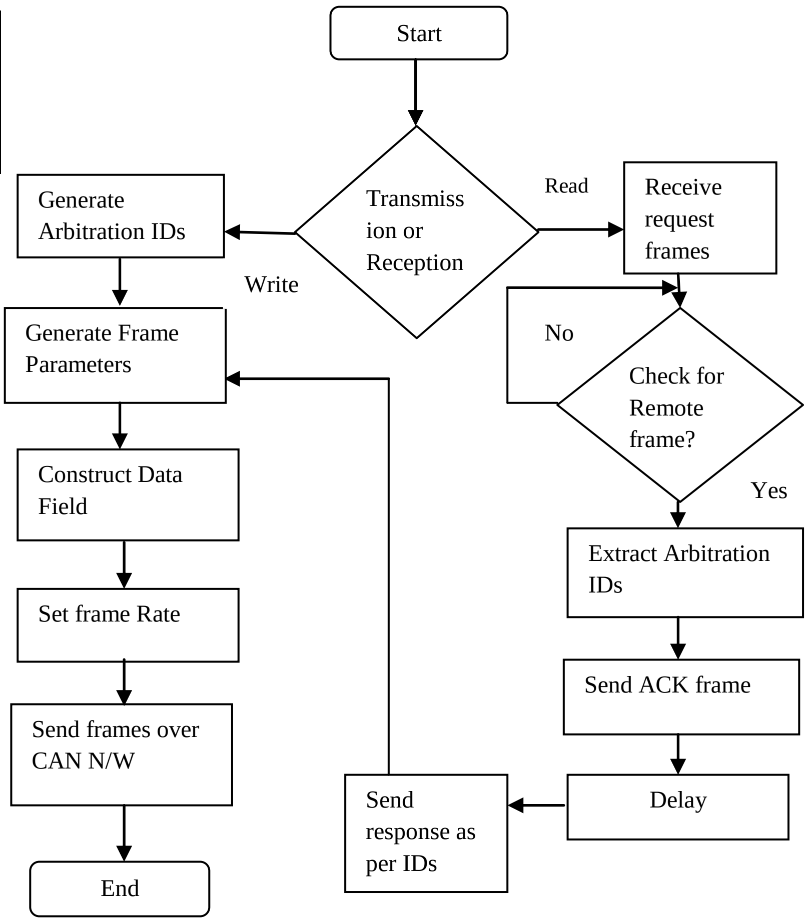


Figure 5: Architecture of Proposed Method

3.1.1 *Transmission of Frames*

The CAN frame transmission means the writing of the messages from simulator to the CAN network through which dashboard is connected. As shown in Figure 5, first configure the CAN port then set the arbitration Id followed by length of data field and frame type that is nothing but the frame parameters. The major part is the data field construction, in this step according to the manufacturing specifications of ECU messages, maintains the byte order of data, width of data and starting bit position. The data can be engine speed, battery voltage, battery current, fuel economy, motor speed etc. It is necessary to maintain the frame rate which can be different for different message frames. It can be achieved by using multiple wait function in LABVIEW. At the end simulator sends the frames over the CAN network through which the digital dashboard is connected.

3.1.2 *Reception of Frames*

Reading of the request frame for particular data from the dashboard to the simulator is nothing but the second part of proposed architecture as shown in Figure 5. For example reception indicates if user wants tire pressure information on dashboard and he selects that option then this will be request frame send to the simulator. In such case the simulator should first send acknowledgement along with values related to tire pressure signal to the digital dashboard. While reading the frame request, first check whether it is remote frame or not. Remote frame is nothing but the request frame for data. Then it is required to extract its arbitration Id to confirm about which message frame has to send as response to that request. Before sending response directly, first send the acknowledgement for the request frame. After that for processing purpose some delay is provided. The response is nothing but transmission of the data frame respective to the requested Id as shown in the architecture.

Proposed simulator can write all data frames sequentially and also it can send the frames on demand, both facilities are provided with this simulator.

4. EXPERIMENTAL RESULTS

To check the functionality of the simulator NI USB CAN module and the PCAN module with LABVIEW platform can be used as shown in Figure 6.

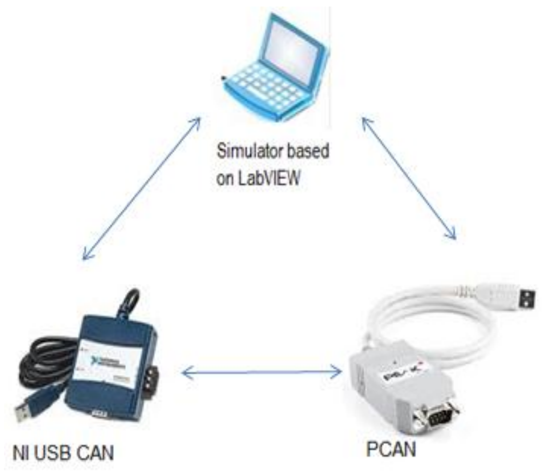


Figure 6: Experimental Set up

The simulator is connected with the tablet via NI CAN USB for testing of the dashboard application. But for experimental purpose we can use the set up shown in Figure 6. Complete simulator design is carried out in LabVIEW platform. The front panels of simulators for two different vehicles are indicated in Figure 7 and Figure 9 as a Simulator A and Simulator B. As the number of ECUs inside vehicle changes then number of CAN messages and associated signals generated by the simulator also get changed.

In Figure 7, one of the parameter is engine oil pressure which is set as 109 KPa and this parameter belong to the Id 18FEE001h. The traced output with PCAN view is as shown in Figure 8, where 6D data of message 18FEE001h is indicated. As $(6D)_{16}=(109)_{10,}$ which confirms the successful transmission of the parameters through simulator to CAN network. Also the cycle time shows the frame rate achieved by the message in Figure 7.

Similarly the functionality of Simulator B can be understood using the Figure 9 and Figure 10. The various parameters belonging to particular message Ids are indicated in Figure10. Total 12 messages with different data fields are shown. Let us take example of battery voltage which is set to value 204V and it is send over 23h Id. It can be traced its transmitted value by observing 023h Id data in Figure 10. On 4th and 5th byte number $(00\ CC)_{16}$ is obtained which is equivalent to the value $(204)_{10}$. Similarly the engine temperature is sending over 281h Id with the value of 112^0 on 4th byte. In Figure 10, value $(00\ 70)_{16}$ ensures the successful transmission of Engine temperature.

To ensure the proper reception of request frames and the appropriate response, PCAN USB view software is used from which it can send the dummy request Id to the simulator via CAN network. As shown in Figure. 12 the 3B0h Id is the remote request send to simulator and its acknowledgement Id 3D0h along with data response belonging to 3E0h Id is send by the simulator. Here when the request of 3B0h arrives to the simulator, it sends the ACK of 3D0h signal after that simulator also sends data frame of 3E0h which consist of data parameters like solar charge power, solar charging voltage and current with the values set as shown in Figure 11. From Figure 12, it is clear that there is successful reception of ACK and data frame from the simulator.

Due to use of these proposed simulators for testing of digital dashboards one can increase the test depths by increasing the values of parameters as this is virtual testing environment.

5. CONCLUSION

In this paper successful development of the CAN based Simulator with LABVIEW is achieved. It is efficiently used for testing of the android application running on the dashboard of the Vehicle. Designing of the writing algorithm of CAN message with accurate periodicity is carried out successfully. Also implementation of receiving of request frames and its response generation for standard as well as extended frame format is completed.

The advantage of virtual test system is mainly the cost reduction. The tests are reproducible for N number of times without loss of data. Increased test depth is achieved with proposed simulator with the help of scaling.

Given simulator is designed with CAN protocol which can be extended to higher level protocols such as J1939 in future. Also the system can be extended by giving generic simulators for various vehicles.

6. ACKNOWLEDGEMENT

The authors would like to acknowledge Mr. Prasanta Sarkar, TATA Motors and TATA Technologies, Pune for their guidance and support.

7. REFERENCES

[1] Steve Corrigan, Introduction to the Controller Area Network (CAN), Texas Instruments, Texas, Application Rep., July 2008.

[2] Controller Area Network (CAN) Overview, National Instruments Co., Austin, Aug 2014.

[3] Vinodh Kumar, Ramesh J, "Automotive In Vehicle Network Protocols," International Conf. on Computer Communication and Informatics, Coimbatore, India, Jan 2014.

[4] Yang Zhu, Hao Hu, Guoqing Xu, Zhiguo Zhao, "Hardware-In-the-Loop Simulation of Pure Electric Vehicle Control System," IEEE Computer Society, 2009.

[5] Uwe Kiencke, Lars Nielsen, Automotive Control Systems For Engine, Driveline, and Vehicle, 2nd ed., Springer, New York, pp. 99-177.

[6] CAN, NI-CANTM Hardware and Software Manual, National Instruments Co., Austin, July 2013.

[7] CAN Physical layer for industrial applications, CAN in Automation (CiA), version 3.0.0, February 2010.

[8] G. Lipovszki, P. Aradi "SIMULATING COMPLEX SYSTEMS AND PROCESSES IN LabVIEW," Journal of Mathematical Sciences, Springer, Vol. 132, No. 5, 2006.

8. APPENDIX

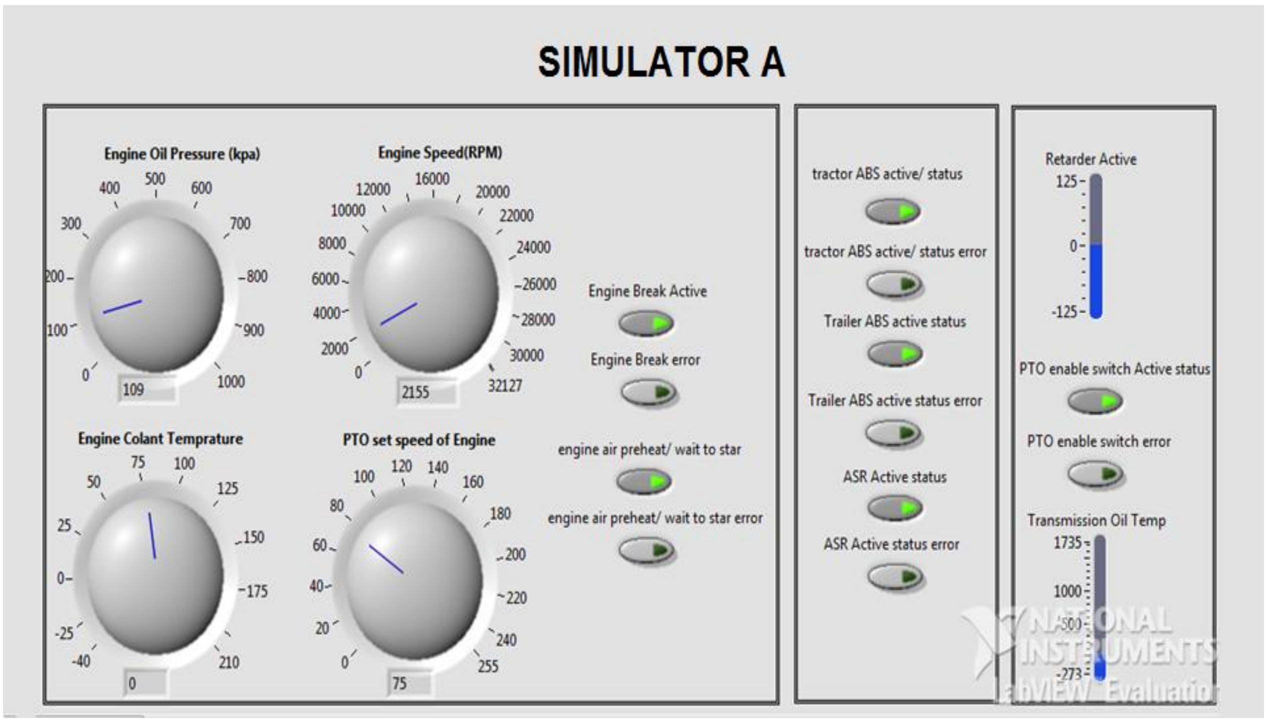
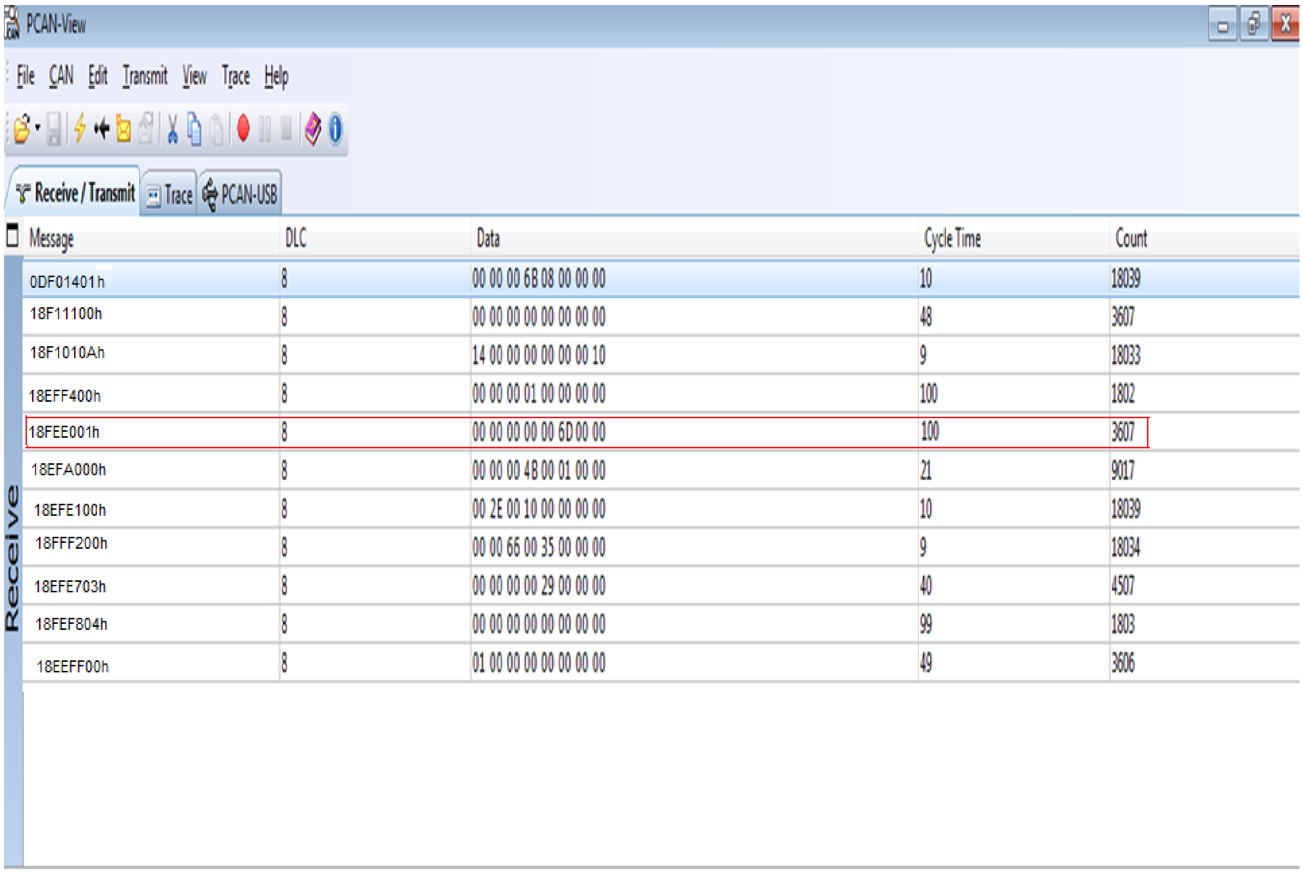

SIMULATOR A
Engine Oil Pressure (kpa)
109
Engine Speed(RPM)
2155
Engine Colant Temprature
0
PTO set speed of Engine
75
Engine Break Active
Engine Break error
engine air preheat/ wait to star
engine air preheat/ wait to star error
tractor ABS active/ status
tractor ABS active/ status error
Trailer ABS active status
Trailer ABS active status error
ASR Active status
ASR Active status error
Retarder Active
PTO enable switch Active status
PTO enable switch error
Transmission Oil Temp
NATIONAL INSTRUMENTS LabVIEW Evaluation

Figure 7: Front Panel of Simulator A for a vehicle



Message	DLC	Data	Cycle Time	Count
0DF01401h	8	00 00 00 6B 08 00 00 00	10	18039
18F11100h	8	00 00 00 00 00 00 00 00	48	3607
18F1010Ah	8	14 00 00 00 00 00 00 10	9	18033
18EFF400h	8	00 00 00 01 00 00 00 00	100	1802
18FEE001h	8	00 00 00 00 00 6D 00 00	100	3607
18EFA000h	8	00 00 00 48 00 01 00 00	21	9017
18EFE100h	8	00 2E 00 10 00 00 00 00	10	18039
18FFF200h	8	00 00 66 00 35 00 00 00	9	18034
18EFE703h	8	00 00 00 00 29 00 00 00	40	4507
18FEF804h	8	00 00 00 00 00 00 00 00	99	1803
18EEFF00h	8	01 00 00 00 00 00 00 00	49	3606

Figure 8: Traced output of simulator A

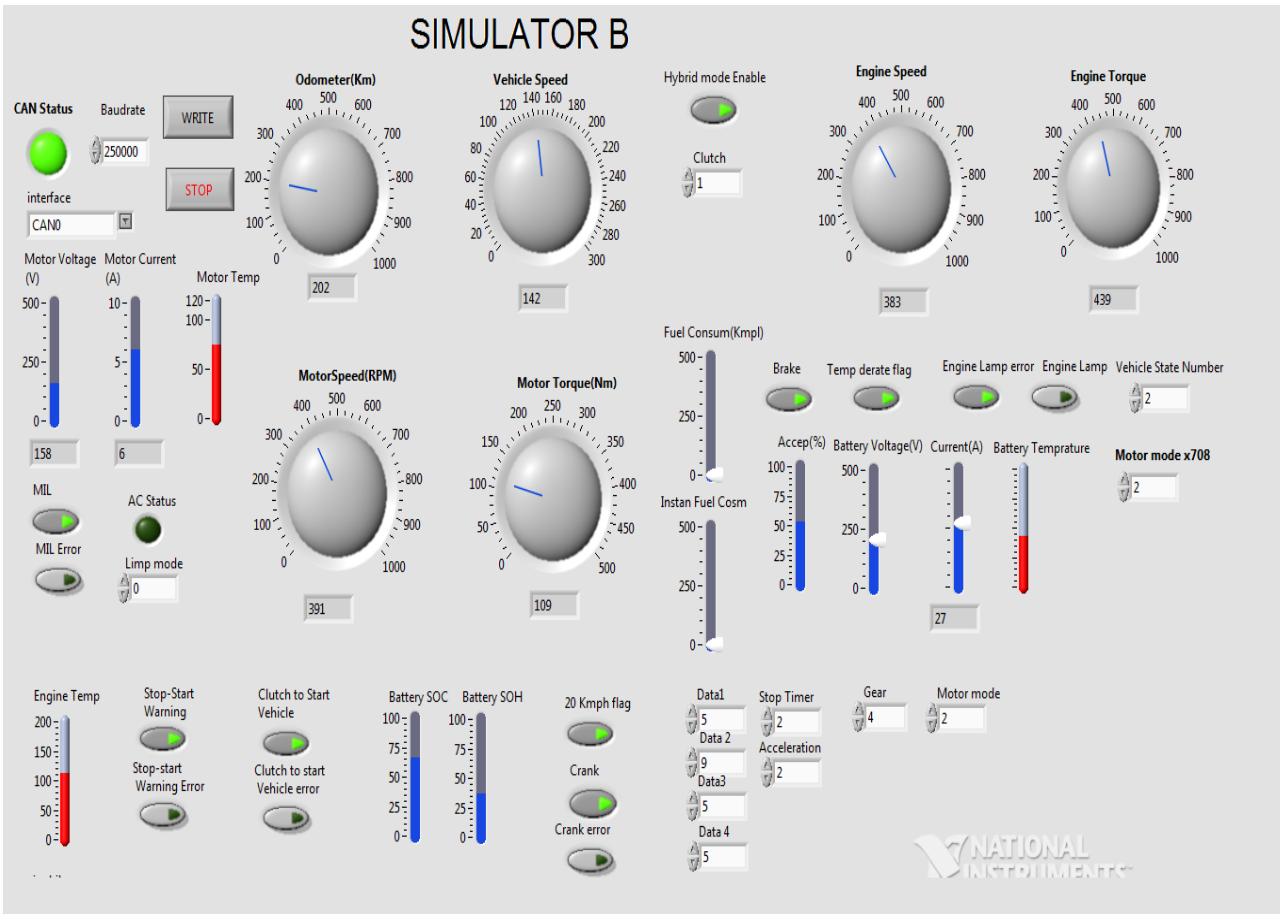
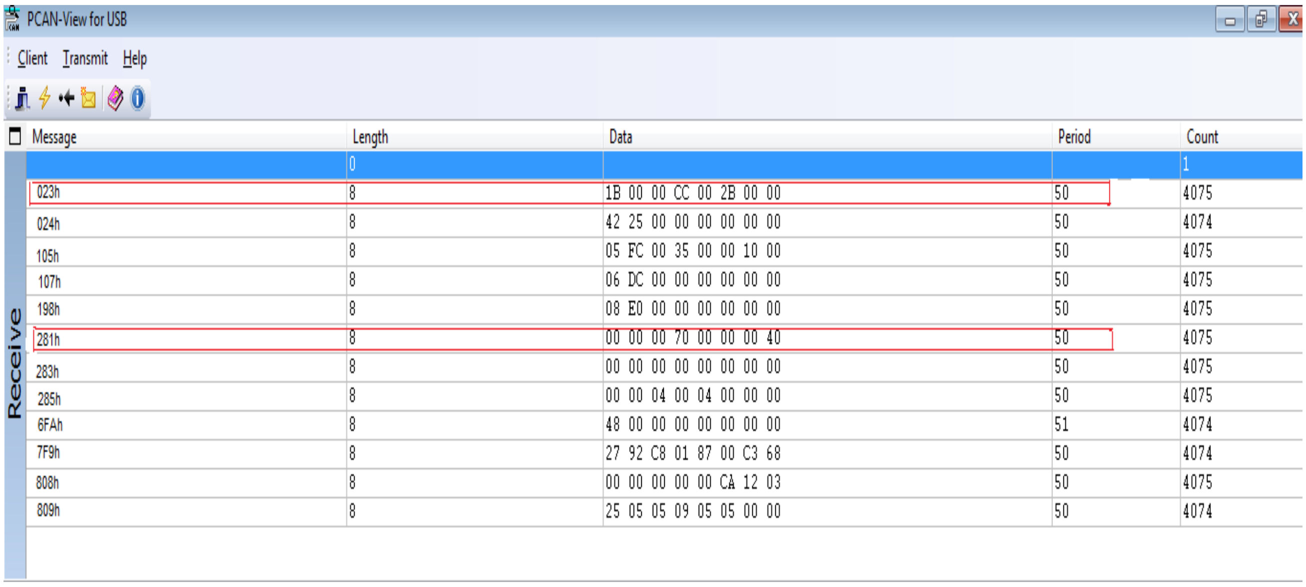
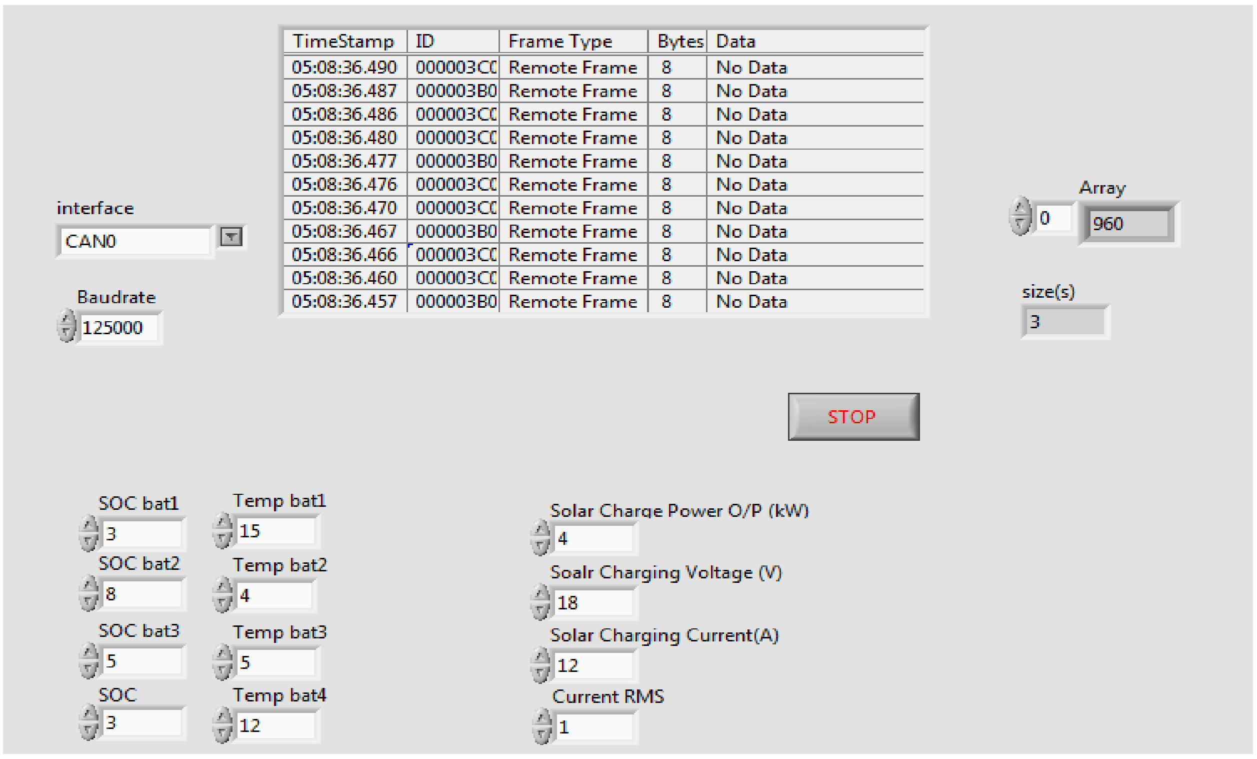


Figure 9: Front Panel of Simulator B for a Vehicle



Message	Length	Data	Period	Count
	0			1
023h	8	1B 00 00 CC 00 2B 00 00	50	4075
024h	8	42 25 00 00 00 00 00 00	50	4074
105h	8	05 FC 00 35 00 00 10 00	50	4075
107h	8	06 DC 00 00 00 00 00 00	50	4075
198h	8	08 E0 00 00 00 00 00 00	50	4075
281h	8	00 00 00 70 00 00 00 40	50	4075
283h	8	00 00 00 00 00 00 00 00	50	4075
285h	8	00 00 04 00 04 00 00 00	50	4075
6FAh	8	48 00 00 00 00 00 00 00	51	4074
7F9h	8	27 92 C8 01 87 00 C3 68	50	4074
808h	8	00 00 00 00 00 CA 12 03	50	4075
809h	8	25 05 05 09 05 05 00 00	50	4074

Figure 10: Traced output of simulator B

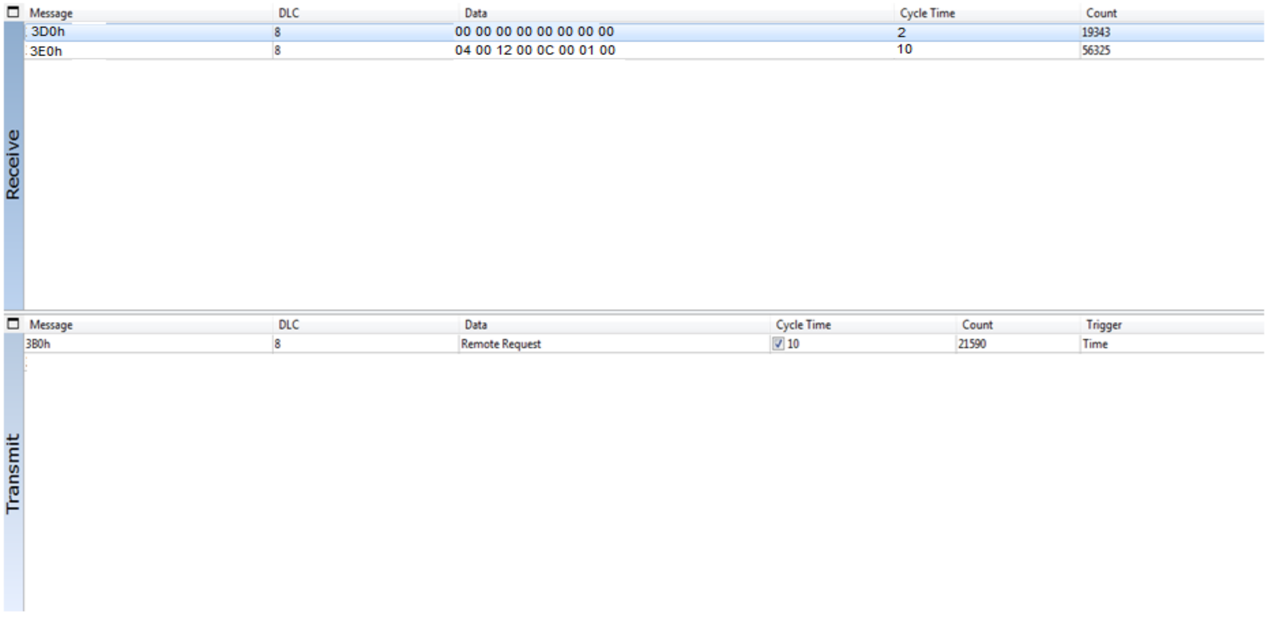


TimeStamp	ID	Frame Type	Bytes	Data
05:08:36.490	000003C0	Remote Frame	8	No Data
05:08:36.487	000003B0	Remote Frame	8	No Data
05:08:36.486	000003C0	Remote Frame	8	No Data
05:08:36.480	000003C0	Remote Frame	8	No Data
05:08:36.477	000003B0	Remote Frame	8	No Data
05:08:36.476	000003C0	Remote Frame	8	No Data
05:08:36.470	000003C0	Remote Frame	8	No Data
05:08:36.467	000003B0	Remote Frame	8	No Data
05:08:36.466	000003C0	Remote Frame	8	No Data
05:08:36.460	000003C0	Remote Frame	8	No Data
05:08:36.457	000003B0	Remote Frame	8	No Data

interface: CAN0
Baudrate: 125000
Array: 0 960
size(s): 3

SOC bat1: 3
Temp bat1: 15
SOC bat2: 8
Temp bat2: 4
SOC bat3: 5
Temp bat3: 5
SOC: 3
Temp bat4: 12
Solar Charge Power O/P (kW): 4
Soalr Charging Voltage (V): 18
Solar Charging Current(A): 12
Current RMS: 1

Figure 11: Front panel of Receiving of frames



Message	DLC	Data	Cycle Time	Count
3D0h	8	00 00 00 00 00 00 00 00	2	19343
3E0h	8	04 00 12 00 0C 00 01 00	10	56325

Message	DLC	Data	Cycle Time	Count	Trigger
3B0h	8	Remote Request	✓ 10	21590	Time

Figure 12: Traced output with PCAN